100

104

Internet

102

client PC

SYN(connect)
SYN-ACK
SYN(get)

⋮

file block

⋮

file block
FIN (close)
ACK
FIN
ACK

TCP listen queue

106

accept →

HTTP listen queue

108

110

httpd daemon + server threads

write

read

I/O controller

116

output buffers

114

112 DISK

FIG. 1

λ_{file}

λ_{tcp}

$\lambda_{tcp} (1-\beta_{tcp})$

$\lambda_{tcp} (1-\beta_{tcp}) (1-\beta_{http})$

$\lambda_{tcp} \pi_{retry} \beta_{tot}$

$\lambda_{tcp} \beta_{tcp}$

$\lambda_{tcp} (1-\beta_{tcp}) \beta_{http}$

$\lambda_{tcp} (1-\pi_{retry}) \beta_{tot}$

$\lambda_{tcp} = \lambda_{file} / (1- \pi_{retry} \beta_{tot})$
$\beta_{tot} = \beta_{tcp} + \beta_{http} - \beta_{tcp} \beta_{http}$

FIG. 2

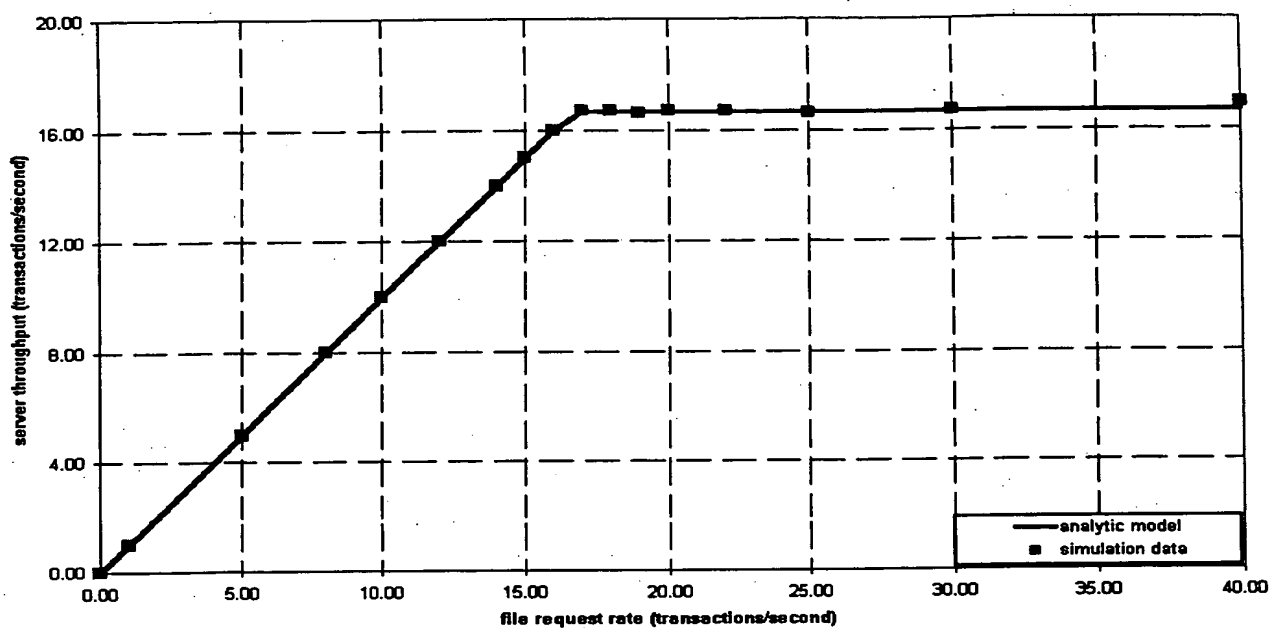**Figure 3.a. Simulated and approximated server thr ughput versus transacti n request rate**

server throughput (transactions/second)

file request rate (transactions/second)

analytic model
simulation data

Figure 3.b. Simulated and approximated end-to-end service time vs. transaction request rate

average delay (seconds)

file request rate (transactions/second)

analytic model
simulation data

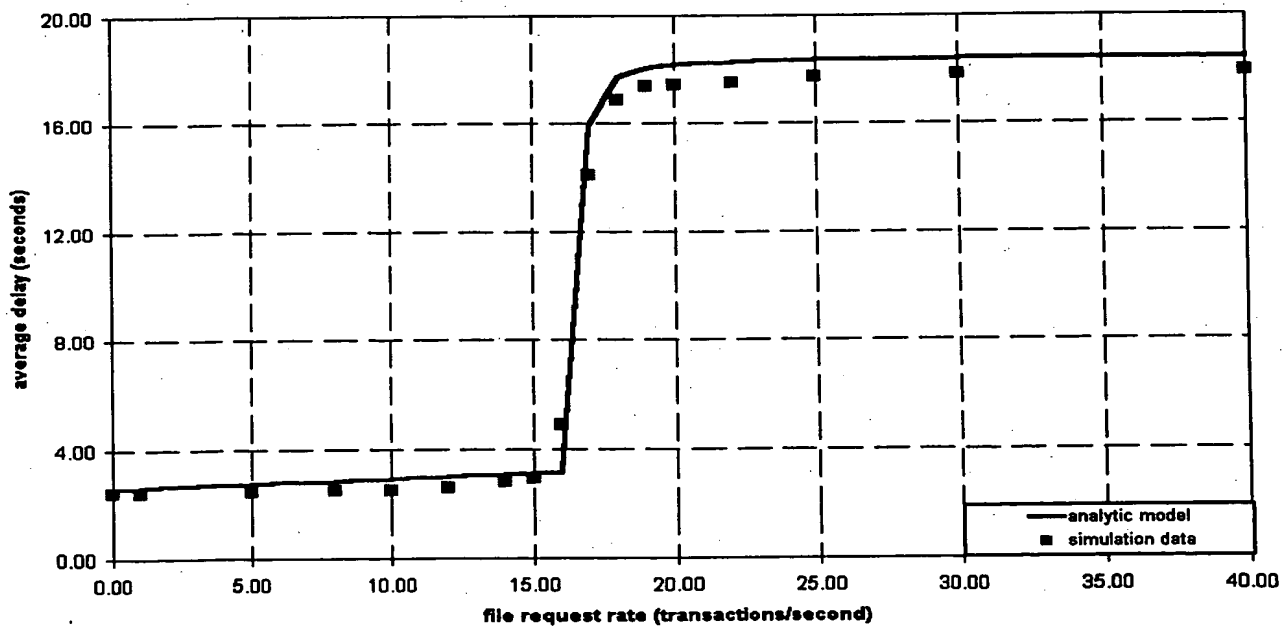**Figure 4.a. Simulated and approximated server throughput versus transaction request rate**

Figure 4.a. Simulated and approximated server throughput versus transaction request rate

y-axis: server throughput (transactions/second)
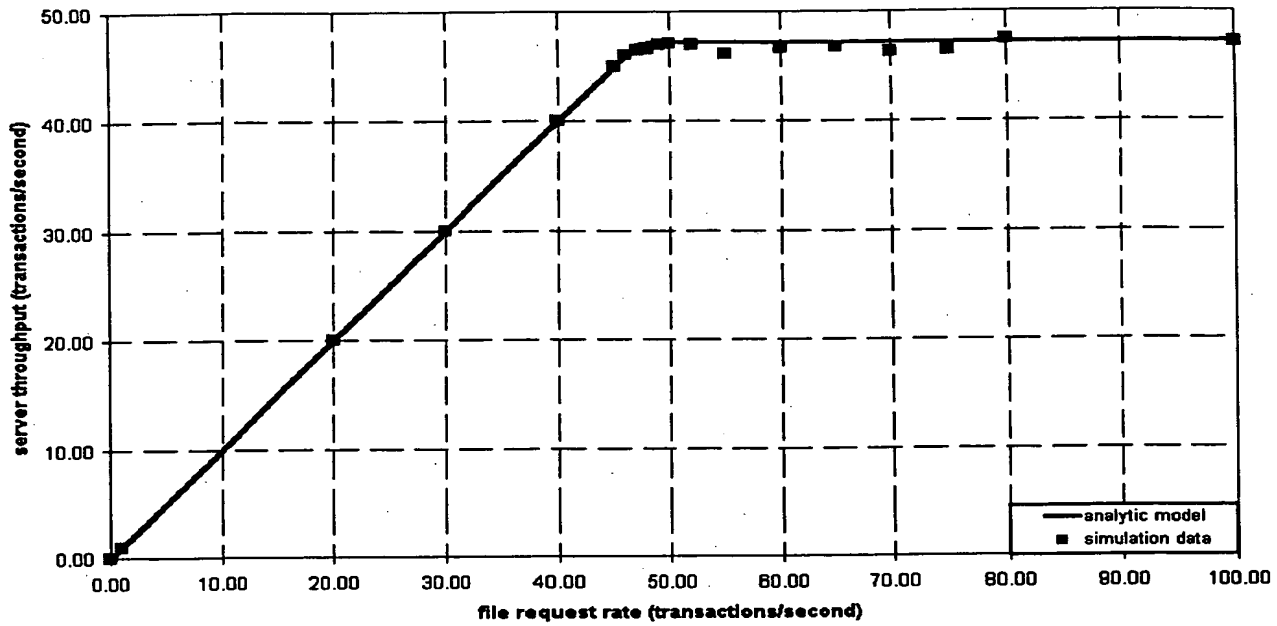x-axis: file request rate (transactions/second)
legend: analytic model, simulation data

Figure 4.b. Simulated and approximated end-to-end service time vs. transaction request rate

Figure 4.b. Simulated and approximated end-to-end service time vs. transaction request rate

y-axis: average delay (seconds)
x-axis: file request rate (transactions/second)
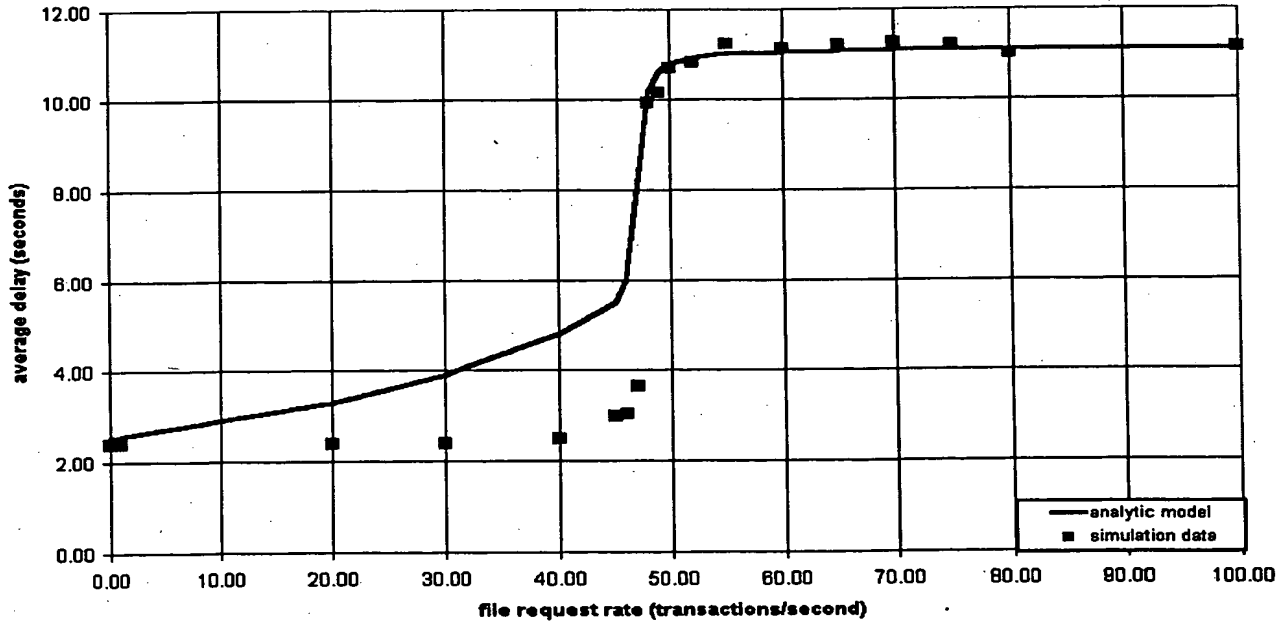legend: analytic model, simulation data

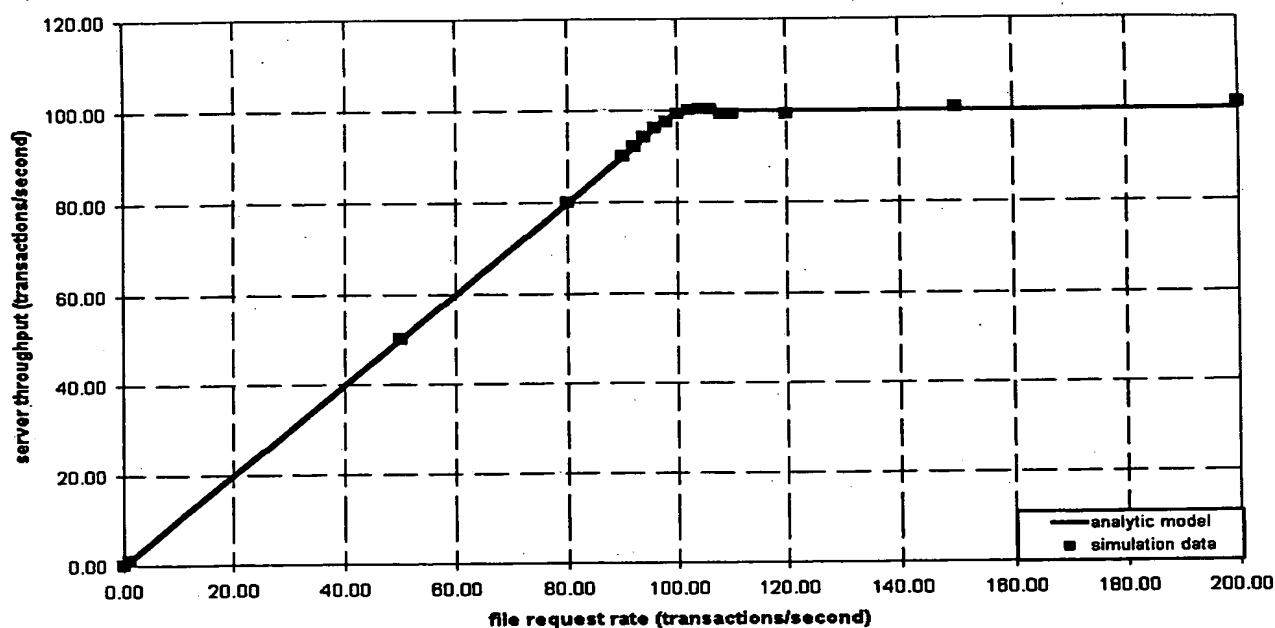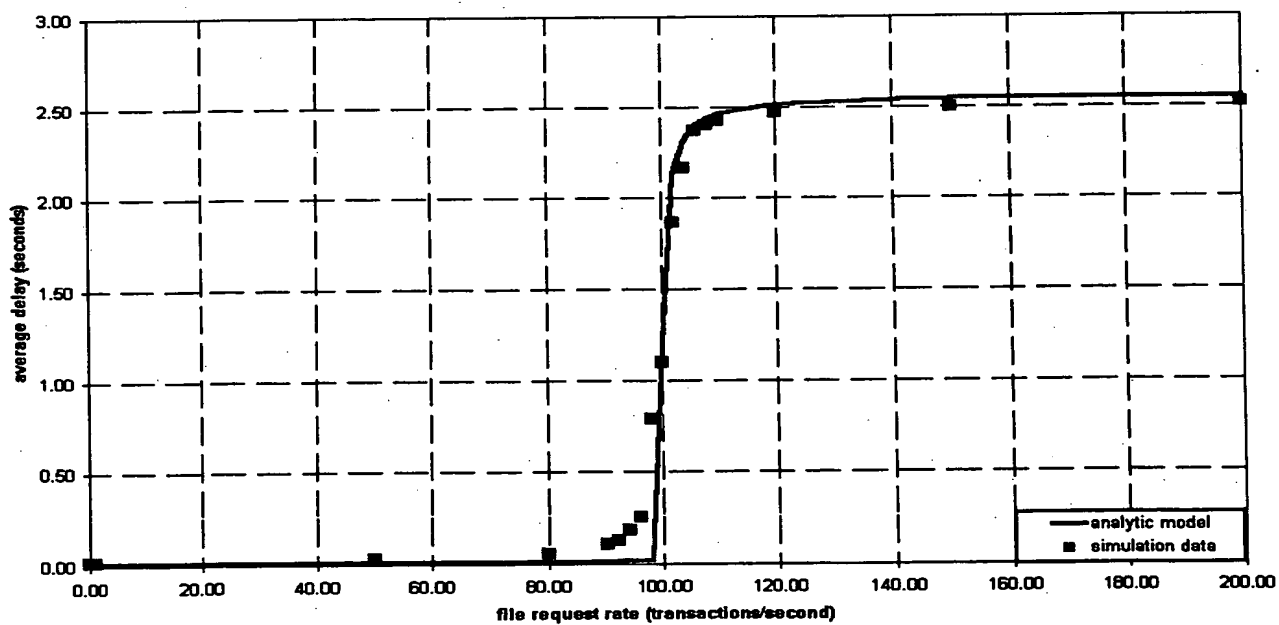**Figure 5.a. Simulated and approximated server throughput versus transacti n request rate**

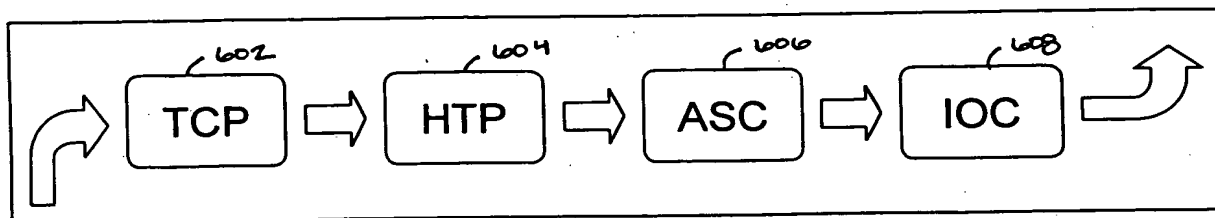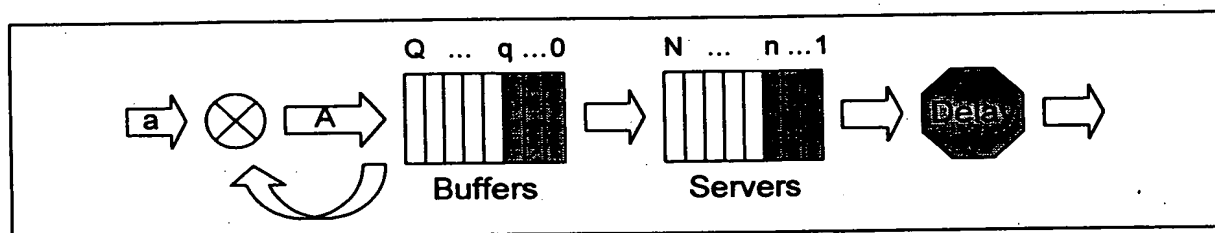**Figure 5.b. Simulated and approximated end-to-end service time vs. transaction request rate**

TCP 602 → HTP 604 → ASC 606 → IOC 608 →

FIG. 6

Q ... q ...0   N ... n ...1

a → ⊗ → A → |Buffers| → |Servers| → Delay →

FIG. 7

800

Start

802
Enter **input** variables
from standard input or file:
a_file, TCP_s, TCP_r,
TCP_Q = 0, TCP_N,
HTP_s, HTP_r, HTP_Q,
HTP_N, ASC_s, ASC_r,
ASC_Q, ASC_N, IOC_s,
IOC_r, IOC_Q = inf, IOC_N

804
Preset output variables:
TCP_p = 0, TCP_b = 0,
HTP_p = 0, HTP_b = 0,
ASC_p = 0, ASC_b = 0,
IOC_p = 0, IOC_b = 0,
TCP_a = HTP_a = a_file,
ASC_a = IOC_a = a_file

806
Compute upper bound IOC
performance by assuming
that no arrivals are blocked
in the TCP, HTP, or ASC
sub-systems: Set
IOC_A = a_file and solve
the system of equations for
IOC_n, IOC_q, IOC_p,
IOC_b, IOC_w

808
Compute upper bound
TCP, HTP, and ASC
performance based on the
value of IOC_w computed
in step A: Set
ASC_S = ASC_s + IOC_w
and solve the equations for
TCP_A, TCP_n, TCP_b,
HTP_A, HTP_n, HTP_q,
HTP_p, HTP_b, HTP_w,
ASC_A, ASC_n, ASC_q,
ASC_p, ASC_b, ASC_w

810
Compute lower bound IOC
performance based on the
values of TCP, HTP, and
ASC blocking computed in
step B: Set
IOC_A = ASC_A(1–ASC_b)
and solve the equations for
IOC_n, IOC_q, IOC_p,
IOC_b, IOC_w

812
Compute lower bound
TCP, HTP, and ASC
performance based on the
value of IOC_w computed
in step C: Set
ASC_S = ASC_s + IOC_w
and solve the equations for
TCP_A, TCP_n, TCP_b,
HTP_A, HTP_n, HTP_q,
HTP_p, HTP_b, HTP_w,
ASC_A, ASC_n, ASC_q,
ASC_p, ASC_b, ASC_w

814
Repeat steps C and D until
the values for all TCP, HTP,
ASC, and IOC performance
measures converge

816
Compute system-wide
output variables:
T_conn = TCP_s + HTP_w,
T_resp = HTP_s + ASC_w
            + ASC_S + IOC_s,
R_conn = IOC_A,
R_error = a_file – IOC_A

END

FIG. 8